

RRRRRRRRRRRR	UUU	UUU	NNN	NNN	000000000	FFFFFFF	FFFFFFF
RRRRRRRRRRRR	UUU	UUU	NNN	NNN	000000000	FFFFFFF	FFFFFFF
RRRRRRRRRRRR	UUU	UUU	NNN	NNN	000000000	FFFFFFF	FFFFFFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNNNNN	NNN	000	FFF	FFF
RRRRRRRRRRRR	UUU	UUU	NNN	NNN	000	FFFFFFF	FFFFFFF
RRRRRRRRRRRR	UUU	UUU	NNN	NNN	000	FFFFFFF	FFFFFFF
RRRRRRRRRRRR	UUU	UUU	NNN	NNN	000	FFFFFFF	FFFFFFF
RRR	RRR	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF

```

AAAAAA LL IIIIII NN NN EEEEEEEEEEE
AAAAAA LL IIIIII NN NN EEEEEEEEEEE
AA AA LL II NN NN EE
AA AA LL II NN NN EE
AA AA LL II NN NN EE
AA AA LL II NN NN EE
AA AA LL II NN NN EE
AAAAAAAA LL II NN NN EEEEEEEEE
AAAAAAAA LL II NN NN EEEEEEEEE
AA AA LL II NN NN EE
AA AA LL II NN NN EE
AA AA LL II NN NN EE
AA AA LLLLLLLLLL IIIIII NN NN EEEEEEEEEEE
AA AA LLLLLLLLLL IIIIII NN NN EEEEEEEEEEE

```

```

LL               IIIIII               SSSSSSSS
LL               IIIIII               SSSSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SSSSSS
LL               II                   SSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS

```



```
0001 0 MODULE aline ( IDENT = 'V04-000'
P 0002 0      %BLISS32 [, ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE,
0003 0      NONEXTERNAL = LONG_RELATIVE) ]
0004 0      ) =
0005 1 BEGIN
0006 1
0007 1
0008 1 *****
0009 1 *
0010 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 *  ALL RIGHTS RESERVED.
0013 1 *
0014 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 *  TRANSFERRED.
0020 1 *
0021 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 *  CORPORATION.
0024 1 *
0025 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1
0032 1 ++
0033 1 FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
0034 1
0035 1 ABSTRACT:      Processes .CENTER, .RIGHT, .LEFT and .INDENT commands.
0036 1
0037 1 ENVIRONMENT:   Transportable
0038 1
0039 1 AUTHOR:        R.W.Friday      CREATION DATE: June, 1978
0040 1
```

ALINE
V04-000

Revision History

F 6
15-Sep-1984 23:46:12
14-Sep-1984 13:05:28

VAX-11 Bliss-32 V4.0-742
DISK\$VMMASTER:[RUNOFF.SRC]ALINE.BLI;1 Page 2 (2)

:	42	0041	1	%SBTTL 'Revision History'
:	43	0042	1	MODIFIED BY:
:	44	0043	1	
:	45	0044	1	008 RER00008 Ron Randall 06-Jun-1983
:	46	0045	1	Centering is now done between left and right text margins.
:	47	0046	1	
:	48	0047	1	007 KFA00007 Ken Alden 16-Mar-1983
:	49	0048	1	PUSH/POP_SCA now visible to dsr.
:	50	0049	1	
:	51	0050	1	006 KFA00006 Ken Alden 07-Mar-1983
:	52	0051	1	Global edit of all modules. Updated module names, idents,
:	53	0052	1	copyright dates. Changed require files to BLISS library.
:	54	0053	1	--
:	55	0054	1	

ALINE
V04-000

Module Level Declarations

6
15-Sep-1984 23:48:12
14-Sep-1984 13:05:28

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]ALINE.BLI;1
Page 3 (3)

ARE

```

: 57      0055 1 %SBTTL 'Module Level Declarations'
: 58      0056 1
: 59      0057 1   INCLUDE FILES:
: 60      0058 1
: 61      0059 1   LIBRARY 'NXPORT:XPORT';           ! XPORT Library
: 62      0060 1   REQUIRE 'REQ:RNODEF';             ! RUNOFF variant definitions
: 63      0191 1
: 64      U 0192 1   %IF DSRPLUS %THEN
: 65      U 0193 1   LIBRARY 'REQ:DPLLIB';           ! DSRPLUS BLISS Library
: 66      0194 1   %ELSE
: 67      0195 1   LIBRARY 'REQ:DSRLIB';             ! DSR BLISS Library
: 68      0196 1   %FI
: 69      0197 1
: 70      0198 1
: 71      0199 1   EQUATED SYMBOLS:
: 72      0200 1
: 73      0201 1   EXTERNAL LITERAL
: 74      0202 1       rintes : UNSIGNED (8);
: 75      0203 1
: 76      0204 1   OWN STORAGE:
: 77      0205 1
: 78      0206 1   OWN
: 79      0207 1       pp_sca : $h_r_sca_block;       ! Used in PUSH_SCA, POP_SCA macros
: 80      0208 1
: 81      0209 1   EXTERNAL REFERENCES:
: 82      0210 1
: 83      0211 1   EXTERNAL
: 84      0212 1       flgt      : flag_table,
: 85      0213 1       gca       : gca_definition,
: 86      0214 1       ira       : FIXED_STRING,
: 87      0215 1       khar,
: 88      0216 1       numprm     : numprm_define,
: 89      0217 1       pdt       : ref_pdt_definition,
: 90      0218 1       sca       : sca_definition,
: 91      0219 1       tsf       : tsf_definition,
: 92      0220 1       ttable    : COUNTED_LIST;
: 93      0221 1
: 94      0222 1   EXTERNAL LITERAL
: 95      0223 1       rnfcjl,      ! Can't justify line
: 96      0224 1       rnfcnf,      ! Character string expected, not found: "<directive>"
: 97      0225 1       rnfinm;      ! Illegal number value: "<directive>"
: 98      0226 1
: 99      0227 1   EXTERNAL ROUTINE
: 100     0228 1       endcmt,
: 101     0229 1       endwrd,
: 102     0230 1       erma,
: 103     0231 1       erml,
: 104     0232 1       guskip,
: 105     0233 1       outnj,
: 106     0234 1       rskips,
: 107     0235 1       scant;
: 108     0236 1
```

```
110 0237 1 GLOBAL ROUTINE aline (handler_code) : NOVALUE =
111 0238 1
112 0239 1 ++
113 0240 1 FUNCTIONAL DESCRIPTION:
114 0241 1
115 0242 1 Processes .CENTER, .RIGHT, .LEFT and .INDENT commands.
116 0243 1
117 0244 1 FORMAL PARAMETERS:
118 0245 1
119 0246 1 handler_code - Indicates which command is to be processed.
120 0247 1
121 0248 1 IMPLICIT INPUTS:
122 0249 1
123 0250 1 numprm - Contains a number, as picked up by GETNUM.
124 0251 1
125 0252 1 IMPLICIT OUTPUTS: None
126 0253 1
127 0254 1 ROUTINE VALUE:
128 0255 1 COMPLETION CODES: None
129 0256 1
130 0257 1 SIDE EFFECTS: None
131 0258 1 --
132 0259 1
133 0260 2 BEGIN
134 0261 2 LOCAL
135 0262 2 hold_tab_count, ! Tab count preserved here.
136 0263 2 text_width, ! Available text width.
137 0264 2 sca_hold : VECTOR [sca_size]; ! SCA preserved here.
138 0265 2
139 0266 2
140 0267 2 ! All these commands accept a number as a parameter.
141 0268 2
142 0269 2 IF NOT .num_result
143 0270 2 THEN
144 0271 2
145 0272 2 ! Ignore the command if the number was bad.
146 0273 2
147 0274 2 RETURN;
148 0275 2
149 0276 2 text_width = .sca_rm - .sca_lm;
150 0277 2
151 0278 2 SELECT .handler_code OF
152 0279 2 SET
153 0280 2
154 0281 2 [h_center] :
155 0282 2
156 0283 2 ! Validate number given in .CENTER command
157 0284 2
158 0285 2 BEGIN
159 0286 2
160 0287 2
161 0288 2 ! Process signed number.
162 0289 2
163 0290 2 IF .num_sign NEQ 0
164 0291 2 THEN
165 0292 2 num_value = .num_value + .sca_rm
166 0293 2
```



```
.. 167      0294      | Process unsigned number.
.. 168      0295
.. 169      0296      ELSE
.. 170      0297
.. 171      0298          IF .num_length EQL 0
.. 172      0299          THEN
.. 173      0300              | Apply default text width.
.. 174      0301              |
.. 175      0302              num_value = .sca_rm;
.. 176      0303
.. 177      0304
.. 178      0305          | Can't center to the left of 0 or beyond the
.. 179      0306          | maximum right margin limit.
.. 180      0307
.. 181      0308          IF (.num_value LEQ 0) OR (.num_value GTR 150)
.. 182      0309          THEN
.. 183      0310              BEGIN
.. 184      0311                  erma (rnfinm, false);
.. 185      0312
.. 186      0313                  | Fix up parameter so text gets centered anyway.
.. 187      0314                  |
.. 188      0315                  num_value = .sca_rm;
.. 189      0316
.. 190      0317              END;
.. 191      0318
.. 192      0319          | Adjust final value to account for later math.
.. 193      0320
.. 194      0321          num_value = .num_value + .sca_lm;
.. 195      0322
.. 196      0323          IF (.num_sign EQL 0) AND (.num_length NEQ 0)
.. 197      0324          THEN
.. 198      0325              num_value = .num_value - .sca_lm;
.. 199      0326
.. 200      0327          END;
.. 201      0328
.. 202      0329      [h_right] :
.. 203      0330          BEGIN
.. 204      0331              | Validate number given on .RIGHT command.
.. 205      0332              | Always handle number as an adjustment to the right margin.
.. 206      0333
.. 207      0334              num_value = .sca_rm - .num_value;
.. 208      0335
.. 209      0336
.. 210      0337          IF (.num_value LSS 0) OR (.num_value GTR 150)
.. 211      0338          THEN
.. 212      0339              | Can't back up to the left of 0 or adjust
.. 213      0340              | beyond the maximum right margin limit.
.. 214      0341              |
.. 215      0342              BEGIN
.. 216      0343                  erma (rnfinm, false);
.. 217      0344
.. 218      0345                  | Fix up parameter so text goes as far right as allowed.
.. 219      0346                  |
.. 220      0347                  num_value = .sca_rm;
.. 221      0348
.. 222      0349              END;
.. 223      0350
```

```

END;
[h_center, h_right] :
BEGIN
    Both .CENTER and .RIGHT can have a ';' or a comment after the
    number, or the text can be on the next line.
    rskips (ira);                ! Skip spaces after the number.
    IF (.khar EQL %C';') OR      ! Just a semi-colon??
        (.flgt [com_flag, flag_enabled] AND ! A comment??
        (.khar EQL .flgt [com_flag, flag_character]))
    THEN
        User said .CENTER/.RIGHT nnn;...
        OR
        .CENTER/.RIGHT nnn!...
        BEGIN
            endcmt ();                ! Skip characters, stopping when
                                      ! a ';' or RINTES is encountered.
        IF .khar NEQ rintes
        THEN
            Scan terminated on a ';', which indicates that
            the text follows on the same line. Skip the ';'
            and verify that there is something there.
            BEGIN
                kcns ();                ! Skip the ';'
            IF .khar EQL rintes
            THEN
                BEGIN
                    A ';' followed by end-of-record is a user error.
                    Since a ';' functions like an end-of-record, what
                    the user has done is give the command an empty record
                    to be .CENTERed or .RIGHTed. The error message warns
                    him of this. Then, processing merges with the normal
                    flow. The result will be that when SCANT gets called
                    it will return a null string, and all the command
                    will do is generate a blank line in the output file.
                    erma (rnfcnf, false);
                END
            END
        END
    ELSE
        END
        IF .khar EQL rintes
        THEN
            !

```



```
281 0408 3      | User said .CENTER/.RIGHT nnn<CR><LF>
282 0409 3
283 0410 4      | BEGIN
284 0411 4
285 0412 4      | Text is on the next record.
286 0413 4
287 0414 4      | EXTERNAL ROUTINE
288 0415 4      |   clh;
289 0416 4
290 0417 4      |   clh (clh_read_input);
291 0418 4      |   kcns ();
292 0419 4      | END
293 0420 4      | ELSE
294 0421 4
295 0422 4      |   Illegal character after the number.
296 0423 4      |   Returning lets DOCM issue the error message.
297 0424 4
298 0425 4      | RETURN;
299 0426 4
300 0427 4
301 0428 4      | At this point it has been ascertained that the .CENTER
302 0429 4      | or .RIGHT command is syntatically correct.
303 0430 4      | Also, the start of the text argument has been located.
304 0431 4
305 0432 4      | Preserve SCA used for normal text processing.
306 0433 4
307 0434 4      | push_sca;                                ! Save special SCA save bits.
308 0435 4
309 0436 4      | INCR I FROM 0 TO (sca_size - 1) DO
310 0437 4      |   sca_hold [.I] = .sca [.I];
311 0438 4
312 0439 4
313 0440 4      | Set up SCA so SCANT preserves white space, and there
314 0441 4      | is lots of room so that a new line doesn't get started
315 0442 4      | unless the user makes an error.
316 0443 4
317 0444 4      | sca_fill   = false;
318 0445 4      | sca_justify = false;
319 0446 4      | sca_lm = 0;
320 0447 4      | sca_rm = 150;
321 0448 4
322 0449 4      | Preserve tab count; temporarily set it to zero
323 0450 4      | so that tabs get replaced by spaces.
324 0451 4
325 0452 4      | hold_tab_count = .ttable [cl_index];
326 0453 4      | ttable [cl_index] = 0;
327 0454 4      | rskips (ira);
328 0455 4      | scant ();                                ! Skip leading spaces and tabs.
329 0456 4      |                                         ! Scan one input line.
330 0457 4
331 0458 4
332 0459 4      | Trailing spaces are dropped, unless there is at least
333 0460 4      | one underlined space in the sequence.
334 0461 4      | Sca_wrd_cpend is not equal to RINTES iff the last character
335 0462 4      | has not yet been made part of the word that was interrupted
336 0463 4      | by the end of the line occurring; this can happen iff
337 0464 4      | trailing spaces/tabs did not force the end of word processing
338 0465 4      | to take place, i.e., there were no trailing spaces/tabs.
```

```

338 0465 3 | See SCANT to see how spaces/tabs get handled.
339 0466 3 |
340 0467 3 | IF .sca_wrd_cpend EQL rintes
341 0468 3 | THEN
342 0469 3 |
343 0470 3 |     IF .sca_wrd_lst_und EQL 0
344 0471 3 |     THEN
345 0472 3 |         sca_wrd_lst_sp = 0;
346 0473 3 |
347 0474 3 |
348 0475 3 |     The call on ENDWRD is made here, rather than letting
349 0476 3 |     OUTNJ do it. The reason it's done here is so that
350 0477 3 |     tsf_ext_hl gets updated, so that the length of the text
351 0478 3 |     can be used.
352 0479 3 |
353 0480 3 | endwrld (false, false, false);
354 0481 3 |
355 0482 3 |
356 0483 3 |     Check to see that the text retrieved is not too long.
357 0484 3 |
358 0485 3 | IF .tsf_ext_hl GTR .text_width
359 0486 3 | THEN
360 0487 3 |     BEGIN
361 0488 3 |     |
362 0489 3 |     | Text is too long even before adjustment.
363 0490 3 |     |
364 0491 3 |     | erml (rnfcjl);
365 0492 3 |     |
366 0493 3 |     | Fix up tsf_ext_hl. However, the entire line
367 0494 3 |     | will still get printed.
368 0495 3 |     |
369 0496 3 |     | tsf_ext_hl = 0;
370 0497 3 |     | END;
371 0498 3 |
372 0499 3 | END;
373 0500 3 |
374 0501 3 | [h_center] :
375 0502 3 | BEGIN
376 0503 3 | |
377 0504 3 | | Compute number of spaces needed to center the line.
378 0505 3 | |
379 0506 3 | | tsf_adjust = (.num_value - .tsf_ext_hl) / 2;
380 0507 3 | | END;
381 0508 3 |
382 0509 3 | [h_right] :
383 0510 3 | BEGIN
384 0511 3 | |
385 0512 3 | | Compute number of spaces needed to push the line the
386 0513 3 | | specified amount to the right.
387 0514 3 | |
388 0515 3 | | tsf_adjust = .num_value - .tsf_ext_hl;
389 0516 3 | | END;
390 0517 3 |
391 0518 3 | [h_center, h_right] :
392 0519 3 | BEGIN
393 0520 3 |
394 0521 3 | IF .tsf_adjust LSS 0
```


ALINE
V04-000

Module Level Declarations

M 6
15-Sep-1984 23:48:12
14-Sep-1984 13:05:28

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]ALINE.BLI;1 Page 9 (4)

```

395      0522 3      THEN
396      0523 4      BEGIN
397      0524 4
398      0525 4      It's not possible to adjust the line.
399      0526 4
400      0527 4      erml (rnfcjl);
401      0528 4
402      0529 4      By setting tsf_adjust to zero, the line
403      0530 4      will go out against the left edge of the page.
404      0531 4
405      0532 4      tsf_adjust = 0;
406      0533 4      END;
407      0534 4
408      0535 4
409      0536 4      If, after going through all the above, it turns
410      0537 4      out to be the case that no text was picked up,
411      0538 4      skip a blank line, unconditionally.
412      0539 4
413      0540 4      IF .tsf_int_hl EQL 0
414      0541 4      THEN
415      0542 4          guskip (1);
416      0543 4
417      0544 4      outnj ();          ! Force out line of text.
418      0545 4
419      0546 4      Restore tab count and SCA to previous status.
420      0547 4
421      0548 4      ttable [cl_index] = .hold_tab_count;
422      0549 4
423      0550 4
424      0551 4      When restoring SCA note that case rules 'play through'.
425      0552 4
426      0553 4      INCR I FROM sca_case_size TO (sca_size - 1) DO
427      0554 4          sca [.I] = .sca_hold [.I];
428      0555 4
429      0556 4      pop_sca;          ! Restore the special SCA bits.
430      0557 4      sca_sect_empty = false;    ! There's something in this section.
431      0558 4      RETURN;
432      0559 4      END;
433      0560 4
434      0561 4      [h_left, h_indent] :
435      0562 4      BEGIN
436      0563 4
437      0564 4
438      0565 4      The specified indentation is just remembered.
439      0566 4      It will be picked up by FCIMRA when the first character for the
440      0567 4      next line has been picked up.
441      0568 4
442      0569 4      IF .num_length NEQ 0
443      0570 4      THEN
444      0571 4          BEGIN          ! A number was supplied.
445      0572 4              sca_indent = .num_value;
446      0573 4          END
447      0574 4      ELSE
448      0575 4          BEGIN          ! A number was not supplied.
449      0576 4              sca_indent = .pdt_indent;    ! Use paragraph indentation
450      0577 4          END;
451      0578 4
```

ALINE
V04-000

Module Level Declarations

: 452
: 453
: 454
: 455
: 456

0579 2 END;
0580 2
0581 2 TES;
0582 2
0583 1 END;

N 6
15-Sep-1984 23:48:12
14-Sep-1984 13:05:28

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[RUNOFF.SRC]ALINE.BLI;1 Page 10 (4)

! End of ALINE

.TITLE ALINE
.IDENT \V04-000\

.PSECT \$OWN\$,NOEXE,2

00000 PP_SCA: .BLKB 48

.EXTRN RINTES, FLGT, GCA
.EXTRN IRA, KHAR, NUMPRM
.EXTRN PDT, SCA, TSF, TTABLE
.EXTRN RNFCJL, RNFCNF, RNFINM
.EXTRN ENDCMT, ENDWRD, ERMA
.EXTRN ERML, GUSKIP, OUTNJ
.EXTRN RSKIPS, SCANT, CLH

.PSECT \$CODE\$,NOWRT,2

			OFFC 00000	.ENTRY ALINE, Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 0237
5B	00G	8F	9A 00002	MOVZBL #RINTES, R11	
5A	00000000G	EF	9E 00006	MOVAB TSF, R10	
59	00000000G	EF	9E 0000D	MOVAB KHAR, R9	
58	00000000G	EF	9E 00014	MOVAB IRA+12, R8	
57	00000000G	EF	9E 0001B	MOVAB NUMPRM+4, R7	
56	00000000G	EF	9E 00022	MOVAB PP_SCA, R6	
55	00000000G	EF	9E 00029	MOVAB SCA+120, R5	
5E	FE80	CE	9E 00030	MOVAB -384(SP), SP	
01	FC	A7	E8 00035	BLBS NUMPRM, 1\$	: 0269
			04 00039	RET	
50	00	B5	D0 0003A 1\$:	MOVL @SCA+120, R0	: 0276
50	FC	B5	C3 0003E	SUBL3 @SCA+116, R0, TEXT_WIDTH	
53	04	AC	D0 00043	MOVL HANDLER_CODE, R3	: 0278
0C		53	D1 00047	CMPL R3, #12	: 0281
		45	12 0004A	BNEQ 6\$	
	04	A7	D5 0004C	TSTL NUMPRM+8	: 0290
		05	13 0004F	BEQL 2\$	
67		50	C0 00051	ADDL2 R0, NUMPRM+4	: 0292
		08	11 00054	BRB 3\$	
	08	A7	D5 00056 2\$:	TSTL NUMPRM+12	: 0298
		03	12 00059	BNEQ 3\$	
67		50	D0 0005B	MOVL R0, NUMPRM+4	: 0303
50		67	D0 0005E 3\$:	MOVL NUMPRM+4, R0	: 0309
		09	15 00061	BLEQ 4\$	
00000096	8F	50	D1 00063	CMPL R0, #150	
		13	15 0006A	BLEQ 5\$	
		7E	D4 0006C 4\$:	CLRL -(SP)	: 0312
	00000000G	8F	DD 0006E	PUSHL #RNFINM	
		02	FB 00074	CALLS #2, ERMA	
67	00	B5	D0 0007B	MOVL @SCA+120, NUMPRM+4	: 0316
67	FC	B5	C0 0007F 5\$:	ADDL2 @SCA+116, NUMPRM+4	: 0322
	04	A7	D5 00083	TSTL NUMPRM+8	: 0324

			09	12	00086	BNEQ	6\$		
		08	A7	D5	00088	TSTL	NUMPRM+12		
			04	13	0008B	BEQL	6\$		
		FC	B5	C2	0008D	SUBL2	@SCA+116, NUMPRM+4	0326	
000000B0	67		53	D1	00091	6\$:	CMPL	R3, #176	0330
	8F		26	12	00098	BNEQ	8\$		
67	00		67	C3	0009A	SUBL3	NUMPRM+4, @SCA+120, NUMPRM+4	0336	
			67	D0	0009F	MOVL	NUMPRM+4, R0	0338	
			09	19	000A2	BLSS	7\$		
00000096	8F		50	D1	000A4	CMPL	R0, #150		
			13	15	000AB	BLEQ	8\$		
			7E	D4	000AD	7\$:	CLRL	-(SP)	0345
		00000000G	8F	DD	000AF	PUSHL	#RNFIM		
00000000G	EF		02	FB	000B5	CALLS	#2, ERMA		
	67		B5	D0	000BC	MOVL	@SCA+120, NUMPRM+4	0349	
	0C		53	D1	000C0	8\$:	CMPL	R3, #12	0354
			0C	13	000C3	BEQL	9\$		
000000B0	8F		53	D1	000C5	CMPL	R3, #176		
			03	13	000CC	BEQL	9\$		
			012C	31	000CE	BRW	19\$		
		F4	A8	9F	000D1	9\$:	PUSHAB	IRA	0360
00000000G	EF		01	FB	000D4	CALLS	#1, RSKIPS		
	52		69	D0	000DB	MOVL	KHAR, R2	0362	
	3B		52	D1	000DE	CMPL	R2, #59		
			10	13	000E1	BEQL	10\$		
		00000000G	EF	E9	000E3	BLBC	FLGT+56, 13\$	0363	
00000000G	40		52	D1	000EA	CMPL	R2, FLGT+128	0364	
	EF		37	12	000F1	BNEQ	13\$		
00000000G	EF		00	FB	000F3	10\$:	CALLS	#0, ENDCMT	0371
	5B		69	D1	000FA	CMPL	KHAR, R11	0374	
			4F	13	000FD	BEQL	16\$		
			68	D5	000FF	TSTL	IRA+12	0382	
			08	14	00101	BGTR	11\$		
	69		5B	9A	00103	MOVZBL	R11, KHAR		
	68		01	CE	00106	MNEGL	#1, IRA+12		
			09	11	00109	BRB	12\$		
	69		B8	9A	0010B	11\$:	MOVZBL	@IRA+4, KHAR	
		F8	A8	D6	0010F	INCL	IRA+4		
		F8	68	D7	00112	DECL	IRA+12		
	5B		69	D1	00114	12\$:	CMPL	KHAR, R11	0384
			35	12	00117	BNEQ	16\$		
			7E	D4	00119	CLRL	-(SP)	0397	
		00000000G	8F	DD	0011B	PUSHL	#RNFIM		
00000000G	EF		02	FB	00121	CALLS	#2, ERMA		
			24	11	00128	BRB	16\$	0374	
	5B		52	D1	0012A	13\$:	CMPL	R2, R11	0405
			01	13	0012D	BEQL	14\$		
				04	0012F	RET			
			05	DD	00130	14\$:	PUSHL	#5	0417
00000000G	EF		01	FB	00132	CALLS	#1, CLH		
			68	D5	00139	TSTL	IRA+12	0418	
			08	14	0013B	BGTR	15\$		
	69		5B	9A	0013D	MOVZBL	R11, KHAR		
	68		01	CE	00140	MNEGL	#1, IRA+12		
			09	11	00143	BRB	16\$		
	69		B8	9A	00145	15\$:	MOVZBL	@IRA+4, KHAR	
		F8	A8	D6	00149	INCL	IRA+4		

			66	EC	68	D7	0014C	DECL	IRA+12		
			A6	F0	B5	D0	0014E	16\$:	MOV	ASCA+100, PP_SCA	0425
04			A6	F4	B5	D0	00152		MOV	ASCA+104, PP_SCA+4	
08			A6	F8	B5	D0	00157		MOV	ASCA+108, PP_SCA+8	
0C			A6	FC	B5	D0	0015C		MOV	ASCA+112, PP_SCA+12	
10			A6	00	B5	D0	00161		MOV	ASCA+116, PP_SCA+16	
14			A6	04	B5	D0	00166		MOV	ASCA+120, PP_SCA+20	
18			A6	08	B5	D0	0016B		MOV	ASCA+124, PP_SCA+24	
1C			A6	0C	B5	D0	00170		MOV	ASCA+128, PP_SCA+28	
20			A6	10	B5	D0	00175		MOV	ASCA+132, PP_SCA+32	
24			A6	14	B5	D0	0017A		MOV	ASCA+136, PP_SCA+36	
28			A6	18	B5	D0	0017F		MOV	ASCA+140, PP_SCA+40	
2C			A6	50	D4	00184	00189		MOV	ASCA+144, PP_SCA+44	
				88	A5	40	0018B	17\$:	CLRL	I	0436
F2		6E40	50	0000005F	8F	F3	00191		MOV	SCA[I], SCA_HOLD[I]	0437
				F0	B5	D4	00199		ACBLEQ	#95, I, 17\$	
				EC	B5	D4	0019C		CLRL	ASCA+104	0444
				FC	B5	D4	0019F		CLRL	ASCA+100	0445
				96	8F	9A	001A2		CLRL	ASCA+116	0446
	00	B5	52	00000000G	EF	D0	001A7		MOVZBL	#150, ASCA+120	0447
				00000000G	EF	D4	001AE		MOV	TTABLE+4, HOLD_TAB_COUNT	0452
				F4	A8	9F	001B4		CLRL	TTABLE+4	0453
	00000000G	EF			01	FB	001B7		PUSHAB	IRA	0454
	00000000G	EF			00	FB	001BE		CALLS	#1, RSKIPS	
		5B		00A0	C5	D1	001C5		CALLS	#0, SCANT	0455
					0A	12	001CA		CMPL	SCA+280, R11	0467
				00DC	C5	D5	001CC		BNEQ	18\$	
					04	12	001D0		TSTL	SCA+340	0470
				00D4	C5	D4	001D2		BNEQ	18\$	
					7E	7C	001D6	18\$:	CLRL	SCA+332	0472
					7E	D4	001D8		CLRL	-(SP)	0480
	00000000G	EF			03	FB	001DA		CLRL	-(SP)	
		50			6A	D0	001E1		CALLS	#3, ENDWRD	
		54		04	A0	D1	001E4		MOV	TSF, R0	0485
					13	15	001E8		CMPL	4(R0), TEXT_WIDTH	
				00000000G	8F	DD	001EA		BLEQ	19\$	
	00000000G	EF			01	FB	001F0		PUSHL	#RNFCJL	0491
		50			6A	D0	001F7		CALLS	#1, ERML	
				04	A0	D4	001FA		MOV	TSF, R0	
		0C			53	D1	001FD	19\$:	CLRL	4(R0)	0496
					0D	12	00200		CMPL	R3, #12	0501
		50			6A	D0	00202		BNEQ	20\$	
		67		04	A0	C3	00205		MOV	TSF, R0	0502
28	51				02	C7	0020A		SUBL3	4(R0), NUMPRM+4, R1	0506
	A0				53	D1	0020F	20\$:	DIVL3	#2, R1, 40(R0)	
	000000B0	8F			09	12	00216		CMPL	R3, #176	0509
					6A	D0	00218		BNEQ	21\$	
		50			6A	D0	00218		MOV	TSF, R0	0510
28	A0			04	A0	C3	0021B		SUBL3	4(R0), NUMPRM+4, 40(R0)	0515
		67			53	D1	00221	21\$:	CMPL	R3, #12	0518
		0C			0C	13	00224		BEQL	22\$	
	000000B0	8F			53	D1	00226		CMPL	R3, #176	
					03	13	0022D		BEQL	22\$	
					0087	31	0022F		BRW	26\$	
		50			6A	D0	00232	22\$:	MOV	TSF, R0	0521
			28		A0	D5	00235		TSTL	40(R0)	
					13	18	00238		BGEQ	23\$	


```
00000000G 8F DD 0023A      PUSHL  #RNFCJL      : 0527
00000000G EF 01 FB 00240      CALLS  #1, ERML      :
50          6A D0 00247      MOVL    TSF, R0      :
          28 A0 D4 0024A      CLRL    40(R0)      : 0532
          00 BA D5 0024D 23$: TSTL    @TSF      : 0540
          09 12 00250      BNEQ     24$      :
          01 DD 00252      PUSHL  #1      : 0542
          01 FB 00254      CALLS  #1, GUSKIP      :
          00 FB 0025B 24$: CALLS  #0, OUTNJ      : 0544
          52 D0 00262      MOVL    HOLD_TAB_COUNT, TTABLE+4 : 0548
          19 D0 00269      MOVL    #25, -1      : 0553
          6E 40 D0 0026C 25$: MOVL    SCA_HOLD[I], SCA[I] : 0554
          8F F3 00272      AOBLEQ  #95, I, 25$      :
          66 D0 0027A      MOVL    PP_SCA, @SCA+100      :
          04 A6 D0 0027E      MOVL    PP_SCA+4, @SCA+104      :
          08 A6 D0 00283      MOVL    PP_SCA+8, @SCA+108      :
          0C A6 D0 00288      MOVL    PP_SCA+12, @SCA+112      :
          10 A6 D0 0028D      MOVL    PP_SCA+16, @SCA+116      :
          14 A6 D0 00292      MOVL    PP_SCA+20, @SCA+120      :
          18 A6 D0 00297      MOVL    PP_SCA+24, @SCA+124      :
          1C A6 D0 0029C      MOVL    PP_SCA+28, @SCA+128      :
          20 A6 D0 002A1      MOVL    PP_SCA+32, @SCA+132      :
          24 A6 D0 002A6      MOVL    PP_SCA+36, @SCA+136      :
          28 A6 D0 002AB      MOVL    PP_SCA+40, @SCA+140      :
          2C A6 D0 002B0      MOVL    PP_SCA+44, @SCA+144      :
          3C A5 D4 002B5      CLRL    SCA+180      : 0557
          04 002B8      RET      : 0519
00000067 8F 53 D1 002B9 26$: CMPL    R3, #103      : 0561
          09 13 002C0      BEQL     27$      :
0000006C 8F 53 D1 002C2      CMPL    R3, #108      :
          12 12 002C9      BNEQ     29$      :
          08 A7 D5 002CB 27$: TSTL    NUMPRM+12      : 0569
          05 13 002CE      BEQL     28$      :
          64 A5 67 D0 002D0      MOVL    NUMPRM+4, SCA+220      : 0572
          04 002D4      RET      : 0569
          64 A5 00000000G FF D0 002D5 28$: MOVL    @PDT, SCA+220      : 0576
          04 002DD 29$: RET      : 0583
```

; Routine Size: 734 bytes, Routine Base: \$CODE\$ + 0000

```
: 457      0584 1
: 458      0585 1 END
: 459      0586 0 ELUDOM
```

! End of module

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	48	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	734	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
\$255\$DUA28:[SYSLIB]XPORT.L32;1	590	0	0	252	00:00.2
\$255\$DUA28:[RUNOFF.SRC]DSRLIB.L32;1	1248	71	5	86	00:00.3

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:ALINE/OBJ=OBJ\$:ALINE MSRC\$:ALINE/UPDATE=(ENHS\$:ALINE)

; Size: 734 code + 48 data bytes
; Run Time: 00:15.8
; Elapsed Time: 00:44.1
; Lines/CPU Min: 2232
; Lexemes/CPU-Min: 18979
; Memory Used: 166 pages
; Compilation Complete

0337 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY